

Robust Near-Separable Nonnegative Matrix Factorization Using Linear Optimization

Nicolas Gillis*

ICTEAM Institute

Université catholique de Louvain

B-1348 Louvain-la-Neuve

Email: nicolas.gillis@uclouvain.be

Robert Luce†

Institut für Mathematik

Technische Universität Berlin

Straße des 17. Juni 136 - 10623 Berlin

Email: luce@math.tu-berlin.de

Abstract

Nonnegative matrix factorization (NMF) has been shown recently to be tractable under the *separability assumption*, under which all the columns of the input data matrix belong to the convex cone generated by only a few of these columns. Bittorf, Recht, Ré and Tropp (‘Factoring non-negative matrices with linear programs’, NIPS 2012) proposed a linear programming (LP) model, referred to as Hottopixx, which is robust under any small perturbation of the input matrix. However, Hottopixx has two important drawbacks: (i) the input matrix has to be normalized, and (ii) the factorization rank has to be known in advance. In this paper, we generalize Hottopixx in order to resolve these two drawbacks, that is, we propose a new LP model which does not require normalization and detects the factorization rank automatically. Moreover, the new LP model is more flexible, significantly more tolerant to noise, and can easily be adapted to handle outliers and other noise models. Finally, we show on several synthetic datasets that it outperforms Hottopixx while competing favorably with two state-of-the-art methods.

Keywords. Nonnegative matrix factorization, separability, linear programming, convex optimization, robustness to noise, pure-pixel assumption, hyperspectral unmixing.

1 Introduction

Nonnegative matrix factorization (NMF) is a powerful dimensionality reduction technique as it automatically extracts sparse and meaningful features from a set of nonnegative data vectors: Given n non-negative m -dimensional vectors gathered in a nonnegative matrix $M \in \mathbb{R}_+^{m \times n}$ and a factorization rank r , NMF computes two nonnegative matrices $W \in \mathbb{R}_+^{m \times r}$ and $H \in \mathbb{R}_+^{r \times n}$ such that $M \approx WH$. In this way, the columns of the matrix W form a basis for the columns of M since $M(:, j) \approx \sum_{k=1}^r W(:, k)H(k, j)$ for all j . Moreover, the nonnegativity constraint on the matrices W and H leads these basis elements to represent common localized features appearing in the dataset as no cancellation can happen in the reconstruction of the original data. Unfortunately, NMF is NP-hard in general [14], and highly ill-posed; see [10] and the references therein. However, if the input data matrix M is *r-separable*, that is, if it can be written as

$$M = W [I_r, H'] \Pi,$$

where I_r is the r -by- r identity matrix, $H' \geq 0$ and Π is a permutation matrix, then the problem can be solved in polynomial time [2]. Algebraically, separability means that there exists a rank- r NMF

*NG is a postdoctoral researcher of the fonds de la recherche scientifique (F.R.S.-FNRS). This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office.

†RL is supported by Deutsche Forschungsgemeinschaft, Cluster of Excellence “UniCat”.

$(W, H) \geq 0$ of M where each column of W is equal to some column of M . Geometrically, r -separability means that the cone generated by the columns of M has r extreme rays given by the columns of W . Equivalently, if the columns of M are normalized to sum to one, r -separability means that the convex hull generated by the columns of M has r vertices given by the columns of W ; see, e.g., [13]. The separability assumption is far from being artificial in several applications:

- In text mining, where each column of M corresponds to a word, separability means that, for each topic, there exists a word associated only with that topic; see [2, 3].
- In hyperspectral imaging, where each column of M equals the spectral signature of a pixel, separability means that, for each constitutive material (“endmembers”) present in the image, there exists a pixel containing only that material; see [11] and the references therein. This assumption is referred to as the *pure-pixel assumption*.
- In blind source separation, where each column of M is a signal measure at a given point in time, separability means that, for each source, there exists a point in time where only that source is active; see [5, 6] and the references therein.

Under the separability assumption, NMF reduces to identifying, among the columns of M , the columns of W allowing to reconstruct all columns of M . In fact, given W , the matrix H can be obtained by solving a linear system (or, in the noisy case, a convex optimization problem $\min_{H \geq 0} \|M - WH\|$).

In this paper, we consider the noisy variant of this problem, referred to as *near-separable NMF*:

(Near-Separable NMF) *Given a noisy r -separable matrix $\tilde{M} = M + N$ with $M = WH = W[I_r, H']\Pi$ where W and H' are nonnegative matrices, Π is a permutation matrix and N is the noise, find a set \mathcal{K} of r indices such that $\tilde{M}(:, \mathcal{K}) \approx W$.*

Several algorithms have been proposed to solve this problem [2, 3, 4, 7, 8, 11, 13]. In this paper, our focus is on the linear programming (LP) model proposed by Bittorf, Recht, Ré and Tropp [4] and referred to as *Hottopixx*. It is described in the next section.

Remark 1 (Nonnegativity of \tilde{M}). *In the formulation of near-separable NMF, the input data matrix \tilde{M} is not necessarily nonnegative since there is no restriction on the noise N . In fact, we will only need to assume that the noise is bounded, but otherwise it is arbitrary; see Section 2.*

1.1 Hottopixx, a Linear Programming Model for Near-Separable NMF

A matrix M is r -separable if and only if

$$M = WH = W[I_r, H']\Pi = [W, WH']\Pi = [W, WH'] \underbrace{\begin{pmatrix} I_r & H' \\ 0_{(n-r) \times r} & 0_{(n-r) \times (n-r)} \end{pmatrix}}_{X^0 \in \mathbb{R}_+^{n \times n}} \Pi = MX^0, \quad (1)$$

for some permutation Π and some matrix $H' \geq 0$. The matrix X^0 is an n -by- n nonnegative matrix with $(n - r)$ zero rows such that $M = MX^0$. Assuming the columns of M sum to one, the columns of W and H' have sum to one as well. Based on these observations, Bittorf, Recht, Ré and Tropp [4] proposed to solve the following optimization problem in order to approximately identify the columns of the matrix W among the columns of the matrix $\tilde{M} = M + N$ where $\|N\|_1 = \max_j \|N(:, j)\|_1 \leq \epsilon$:

$$\begin{aligned} \min_{X \in \mathbb{R}_+^{n \times n}} \quad & p^T \text{diag}(X) \\ \text{such that} \quad & \|\tilde{M} - \tilde{M}X\|_1 \leq 2\epsilon, \\ & \text{tr}(X) = r, \\ & X(i, i) \leq 1 \text{ for all } i, \\ & X(i, j) \leq X(i, i) \text{ for all } i, j, \end{aligned} \quad (2)$$

where p is any n -dimensional vector with distinct entries; see Algorithm 1.

Algorithm 1 Hottopixx - Extracting Columns of a Noisy Separable Matrix by Linear Programming

Input: A normalized noisy r -separable matrix $\tilde{M} = WH + N \in \mathbb{R}_+^{m \times n}$, the factorization rank r , the noise level $\|N\|_1 \leq \epsilon$ and a vector $p \in \mathbb{R}^n$ with distinct entries.

Output: A matrix \tilde{W} such that $\tilde{W} \approx W$ (up to permutation).

- 1: Find the optimal solution X^* of (2) where p has distinct entries.
 - 2: Let \mathcal{K} be the index set corresponding to the r largest diagonal entries of X^* .
 - 3: Set $\tilde{W} = \tilde{M}(:, \mathcal{K})$.
-

Intuitively, the LP model¹ (2) assigns a total weight r to the n diagonal entries of the variable X in such a way that \tilde{M} can be well approximated using nonnegative linear combinations of columns of \tilde{M} corresponding to positive diagonal entries of X . Moreover, the weights used in the linear combinations cannot exceed the diagonal entries of X since $X(:, j) \leq \text{diag}(X)$ for all j . There are several drawbacks in using the LP model (2) in practice:

1. The factorization rank r has to be chosen in advance. In practice the true factorization rank is often unknown, and a “good” factorization rank for the application at hand is typically found by trial and error. Therefore the LP above may have to be resolved many times.
2. The columns of the input data matrix have to be normalized in order to sum to one. This may introduce significant distortions in the dataset and lead to poor performance; see [13] where some numerical experiments are presented.
3. The noise level $\|N\|_1 \leq \epsilon$ has to be estimated.
4. One has to solve a rather large optimization problem with n^2 variables, so that the model cannot be used directly for huge-scale problems.

It is important to notice that there is no way to getting rid of both drawbacks 2. and 3. In fact, in the noisy case, the user has to indicate either

- The factorization rank r , and the algorithm should find a subset of r columns of \tilde{M} as close as possible to the columns of W , or
- The noise level ϵ , and the algorithm should try to find the smallest possible subset of columns of \tilde{M} allowing to approximate \tilde{M} up to the required accuracy.

1.2 Contribution and Outline of the Paper

In this paper, we generalize Hottopixx in order to resolve drawbacks 1. and 2. above. More precisely, we propose a new LP model which has the following properties:

- Given the noise level ϵ , it detects the number r of columns of W automatically; see Section 2.
- It can be adapted to dealing with outliers; see Section 3.
- It does not require column normalization; see Section 4.
- It is significantly more tolerant to noise than Hottopixx. In fact, we propose a tight robustness analysis of the new LP model proving its superiority (see Theorems 1 and 2). This is illustrated in Section 5 on several synthetic datasets, where the new LP model is shown to outperform Hottopixx while competing favorably with two state-of-the-art methods, namely the successive projection algorithm (SPA) from [1, 11] and the fast conical hull algorithm (XRAY) from [13].

¹Strictly speaking, (2) is not a linear program but it can be reformulated as one.

2 Detecting the Factorization Rank Automatically

In this section, we analyze the following LP model:

$$\begin{aligned} \min_{X \in \mathbb{R}_+^{n \times n}} \quad & p^T \text{diag}(X) \\ \text{such that} \quad & \|\tilde{M} - \tilde{M}X\|_1 \leq \rho\epsilon, \\ & X(i, i) \leq 1 \text{ for all } i, \\ & X(i, j) \leq X(i, i) \text{ for all } i, j, \end{aligned} \tag{3}$$

where p has *positive* entries and $\rho > 0$ is a parameter. We also analyze the corresponding near-separable NMF algorithm (Algorithm 2) with an emphasis on *robustness*. The LP model (3) is

Algorithm 2 Extracting Columns of a Noisy Separable Matrix by Linear Programming

Input: A normalized noisy r -separable matrix $\tilde{M} = WH + N \in \mathbb{R}_+^{m \times n}$, the noise level $\|N\|_1 \leq \epsilon$, a parameter $\rho > 0$ and a vector $p \in \mathbb{R}^n$ with positive distinct entries.

Output: An m -by- r matrix \tilde{W} such that $\tilde{W} \approx W$ (up to permutation).

- 1: Compute an optimal solution X^* of (3).
 - 2: Let \mathcal{K} be the index set corresponding to the diagonal entries of X^* larger than $1 - \frac{\min(1, \rho)}{2}$.
 - 3: $\tilde{W} = \tilde{M}(:, \mathcal{K})$.
-

exactly the same as (2) except that the constraint $\text{tr}(X) = r$ has been removed, and that there is an additional parameter ρ . Moreover, the vector $p \in \mathbb{R}^n$ in the objective function has to be *positive*, or otherwise any diagonal entry of an optimal solution of (3) corresponding to a negative entry of p will be equal to one (in fact, this reduces the objective function the most while minimizing $\|\tilde{M} - \tilde{M}X\|_1$). A natural value for the parameter ρ is two, as in the original LP model (2), so that the matrix X^0 in Equation (1) identifying the set of columns of \tilde{M} corresponding to the columns of W is feasible. However, the model (3) is feasible for any $\rho \geq 0$ since the identity matrix of dimension n (that is, $X = I_n$) is always feasible. Hence, it is not clear a priori which value of ρ should be chosen. The reason we analyze the LP model (3) for different values of ρ is two-fold:

- First, it shows that the LP model (3) is rather flexible as it is not too sensitive to the right-hand side of the constraint $\|\tilde{M} - \tilde{M}X\|_1 \leq \rho\epsilon$. In other terms, the noise level does not need to be known precisely for the model to make sense. This is a rather desirable property as, in practice, the value of ϵ is typically only known/evaluated approximately.
- Second, we observed that taking ρ smaller than two gives in average significantly better results (see Section 5 for the numerical experiments). Our robustness analysis of Algorithm 2 will suggest that the best choice is to take $\rho = 1$.

In this section, we prove that the LP model (3) allows to identifying approximately the columns of the matrix W among the columns of the matrix \tilde{M} for *any* $\rho > 0$, given that the noise level ϵ is sufficiently small (ϵ will depend on the value ρ); see Theorems 1, 2 and 3.

Before stating the robustness results, let us define the conditioning of a nonnegative matrix W whose columns sum to one:

$$\kappa = \min_{1 \leq k \leq r} \min_{x \in \mathbb{R}_+^{r-1}} \|W(:, k) - W(:, \mathcal{K})x\|_1, \quad \text{where } \mathcal{K} = \{1, 2, \dots, r\} \setminus \{k\},$$

and the matrix W is said to be κ -*robustly conical*. The parameter $0 \leq \kappa \leq 1$ tells us how well the columns of W are spread in the unit simplex. In particular, if $\kappa = 1$, then W contains the identity

matrix as a submatrix (all other entries being zeros) while, if $\kappa = 0$, then at least one of the columns of W belongs to the convex cone generated by the others. Clearly, the better the columns of W are spread across the unit simplex, the less sensitive is the data to noise. For example, $\epsilon < \frac{\kappa}{2}$ is a necessary condition to being able to distinguish the columns of W [9].

2.1 Robustness Analysis without Duplicates and Near Duplicates

In this section, we assume that the columns of W are isolated (that is, there is no duplicate nor near duplicate of the columns of W in the dataset) hence more easily identifiable. This type of margin constraint is typical in machine learning [4], and is equivalent to bounding the entries of H' in the expression $M = W[I_r, H']\Pi$, see Equation (1). In fact, for any $1 \leq k \leq r$ and $h \in \mathbb{R}_+^r$ with $\max_i h(i) \leq \beta \leq 1$, we have that

$$\begin{aligned} \|W(:, k) - Wh\|_1 &= \|(1 - h(k))W(:, k) - W(:, \mathcal{K})h(\mathcal{K})\|_1 \\ &\geq (1 - \beta) \min_{y \in \mathbb{R}_+^{r-1}} \|W(:, k) - W(:, \mathcal{K})y\|_1 \\ &\geq (1 - \beta)\kappa, \end{aligned}$$

where $\mathcal{K} = \{1, 2, \dots, r\} \setminus \{k\}$. Hence $\max_{ij} H'_{ij} \leq \beta$ implies that all data points are at distance at least $(1 - \beta)\kappa$ of any column of W . Under this condition, we have the following robustness result:

Theorem 1. *Suppose $\tilde{M} = M + N$ where the columns of M sum to one, $M = WH$ admits a rank- r separable factorization of the form (1) with $H \geq 0$, $\max_{ij} H'_{ij} \leq \beta \leq 1$ and W κ -robustly conical with $\kappa > 0$, and $\|N\|_1 \leq \epsilon$. If*

$$\epsilon \leq \frac{\kappa(1 - \beta) \min(1, \rho)}{5(\rho + 2)},$$

then Algorithm 2 extracts a matrix $\tilde{W} \in \mathbb{R}^{m \times r}$ satisfying $\|W - \tilde{W}(:, P)\|_1 \leq \epsilon$ for some permutation P .

Proof. See Appendix A. □

Remark 2 (Noiseless case). *When there is no noise (that is, $N = 0$ and $\epsilon = 0$), duplicates and near duplicates are allowed in the dataset; otherwise $\epsilon > 0$ implying that $\beta < 1$ hence the columns of W are isolated.*

Remark 3 (A slightly better bound). *The bound on the allowable noise in Theorem 1 can be slightly improved, so that under the same conditions we can allow a noise level of*

$$\epsilon < \frac{\kappa(1 - \beta) \min(1, \rho)}{4(\rho + 2) + \kappa(1 - \beta) \min(1, \rho)}.$$

However, the scope for substantial improvements is limited, as we will show in Theorem 2.

Remark 4 (Best choice for ρ). *Our analysis suggests that the best value for ρ is one. In fact,*

$$\operatorname{argmax}_{\rho \geq 0} \frac{\min(1, \rho)}{5(\rho + 2)} = 1.$$

In this particular case, the upper bound on the noise level to guarantee recovery is given by $\epsilon \leq \frac{\kappa(1-\beta)}{15}$ while, for $\rho = 2$, we have $\epsilon \leq \frac{\kappa(1-\beta)}{20}$. The choice $\rho = 1$ is also optimal in the same sense for the bound in the previous remark. We will see in Section 5, where we present some numerical experiments, that choosing $\rho = 1$ works remarkably better than $\rho = 2$.

It was proven in [9] that, for Algorithm 1 to extract the columns of W under the same assumptions as in Theorem 1, it is necessary that

$$\epsilon < \frac{\kappa(1-\beta)}{(r-1)(1-\beta)+1} \quad \text{for any } r \geq 3 \text{ and } \beta < 1,$$

while it is sufficient that $\epsilon \leq \frac{\kappa(1-\beta)}{9(r+1)}$. Therefore, if there are no duplicate nor near duplicate of the columns of W in the dataset,

Algorithm 2 is more robust than Hottopixx (Algorithm 1): in fact, unlike Hottopixx, its bound on the noise to guarantee recovery (up to the noise level) is independent of the number of columns of W . Moreover, given the noise level, it detects the number of columns of W automatically.

The reason for the better performance of Algorithm 2 is the following: for most noisy r -separable matrices \tilde{M} , there typically exist matrices X' satisfying the constraints of (3) and such that $\text{tr}(X') < r$; see, e.g., the constructions in [9]. Therefore, the remaining weight $(r - \text{tr}(X'))$ will be assigned by Hottopixx to the diagonal entries of X' corresponding to the smallest entries of p , since the objective is to minimize $p^T \text{diag}(X')$. These entries are unlikely to correspond to columns of W (in particular, if p is chosen by an adversary as in [9]). We observed that when the noise level ϵ increases, $r - \text{tr}(X')$ increases as well, hence it becomes likely that some columns of W will not be identified. This explains why the LP model enforcing the constraint $\text{tr}(X) = r$ is less robust, and why its bound on the noise depends on the factorization rank r . Moreover, the LP (2) is also much more sensitive to the parameter ϵ than the model LP (3):

- For ϵ sufficiently small, it becomes infeasible, while,
- for ϵ too large, the problem described above is worsened: there are matrices X' satisfying the constraints of (3) and such that $\text{tr}(X') \ll r$, hence Hottopixx will perform rather poorly (especially in the worst-case scenario, that is, if the problem is set up by an adversary).

To conclude this section, we prove that the bound on the noise level ϵ to guarantee the recovery of the columns of W by Algorithm 2 given in Theorem 1 is tight up to some constant multiplicative factor.

Theorem 2. *For any fixed $\rho > 0$ and $\beta < 1$, the bound on ϵ in Theorem 1 is tight up to a multiplicative factor. In fact, under the same assumptions on the input matrix \tilde{M} , it is necessary that $\epsilon < \frac{\kappa(1-\beta)\min(1,\rho)}{2\rho}$ for Algorithm 2 to extract a matrix $\tilde{W} \in \mathbb{R}^{m \times r}$ satisfying $\|W - \tilde{W}(:, P)\|_1 \leq \epsilon$ for some permutation P .*

Proof. See Appendix B. □

For example, Theorem 2 implies that, for $\rho = 1$, the bound of Theorem 1 is tight up to a factor $\frac{15}{2}$.

2.2 Robustness Analysis with Duplicates and Near Duplicates

In case there are duplicates and near duplicates in the dataset, it is necessary to apply a post-processing to the solution of (3). In fact, although we can guarantee that there is a subset of the columns of \tilde{M} close to each column of W whose sum of the corresponding diagonal entries of an optimal solution of (3) is large, there is not guarantee that the weight will be concentrated only in one entry. It is then required to apply some post-processing based on the distances between the data points to the solution of (3) (instead of simply picking the r indices corresponding to its largest diagonal entries) in order to obtain a robust algorithm. In particular, using Algorithm 4 to post-process the solution of (2) leads to a more robust algorithm than Hottopixx [9]. Note that pre-processing would also be possible [8, 2].

Therefore, we propose to post-process an optimal solution of (3) with Algorithm 4; see Algorithm 3, for which we can prove the following robustness result:

Theorem 3. Let $M = WH$ be an r -separable matrix whose columns sum to one of the form (1) with $H \geq 0$ and W κ -robustly conical. Let also $\tilde{M} = M + N$ with $\|N\|_1 \leq \epsilon$. If

$$\epsilon < \frac{\omega\kappa}{99(r+1)},$$

where $\omega = \min_{i \neq j} \|W(:, i) - W(:, j)\|_1$, then Algorithm 3 extracts a matrix \tilde{W} such that

$$\|W - \tilde{W}(:, P)\|_1 \leq \delta = 49(r+1)\frac{\epsilon}{\kappa} + 2\epsilon, \quad \text{for some permutation } P.$$

Proof. See Appendix C (for simplicity, we only consider the case $\rho = 2$; the proof can be generalized for other values of $\rho > 0$ in a similar way as in Theorem 1). \square

This robustness result is the same as for the algorithm using the optimal solution of (2) post-processed with Algorithm 4 [9]. Hence, in case there are duplicates and near duplicates in the dataset, we do not know if Algorithm 3 is more robust, although we believe the bound for Algorithm 3 can be improved (in particular, that the dependence in r can be removed), this is a topic for further research.

Algorithm 3 Extracting Columns of a Noisy Separable Matrix by Linear Programming

Input: A normalized r -separable matrix $\tilde{M} = WH + N$, the noise level $\|N\|_1 \leq \epsilon$ and a vector $p \in \mathbb{R}_+^n$ with positive distinct entries.

Output: An m -by- r matrix \tilde{W} such that $\tilde{W} \approx W$ (up to permutation).

- 1: Compute the optimal solution X^* of (3) where $p = e$ is the vector of all ones and $\rho = 2$.
 - 2: $K = \text{post-processing}(\tilde{M}, \text{diag}(X^*), \epsilon)$;
 - 3: $\tilde{W} = \tilde{M}(:, K)$;
-

Remark 5 (Choice of p). Although Theorem 3 requires the entries of the vector p to be all ones, we recommend to take the entries of p distinct, but close to one. This allows the LP (3) to discriminate better between the duplicates hence Algorithm 3 does not necessarily have to enter the post-processing loop. We suggest to use $p(i) \sim 1 + \mathcal{U}(-\sigma, \sigma)$ for all i , where $\sigma \ll 1$ and $\mathcal{U}(a, b)$ is the uniform distribution in the interval $[a, b]$.

3 Handling Outliers

Removing the rank constraint has another advantage: it allows to deal with outliers. If the dataset contains outliers, the corresponding diagonal entries of an optimal solution X^* of (3) will have to be large (since outliers cannot be approximated well with convex combinations of points in the dataset). However, under some reasonable assumptions, outliers are useless to approximate data points, hence off-diagonal entries of the rows of X^* corresponding to outliers will be small. Therefore, one could discriminate between the columns of W and the outliers by looking at the *off-diagonal entries* of X^* . This result is closely related to the one presented in [11, Section 3]. For simplicity, we consider in this section only the case where $\rho = 2$ and assume absence of duplicates and near-duplicates in the dataset; the more general case can be treated in a similar way.

Let the columns of $T \in \mathbb{R}^{m \times t}$ be t outliers added to the separable matrix $W[I_r, H']$ along with some noise to obtain

$$\tilde{M} = M + N \quad \text{where} \quad M = [W, T]H = [W, T, WH'] \Pi \begin{bmatrix} I_r & 0_{r \times t} & H' \\ 0_{t \times r} & I_t & 0_{t \times r} \end{bmatrix} \Pi, \quad (4)$$

which is a noisy r -separable matrix containing t outliers. We propose Algorithm 5 to approximately extract the columns of W among the columns of \tilde{M} .

Algorithm 4 Post-Processing - Clustering Diagonal Entries of X^* [9]

Input: A matrix $\tilde{M} \in \mathbb{R}^{m \times n}$, a vector $x \in \mathbb{R}_+^n$, $\epsilon \geq 0$, and possibly a factorization rank r .

Output: A index set \mathcal{K}^* with r indices so that the columns of $\tilde{M}(:, \mathcal{K}^*)$ are centroids whose corresponding clusters have large weight (the weights of the data points are given by x).

```
1:  $D(i, j) = \|m_i - m_j\|_1$  for  $1 \leq i, j \leq n$ ;
2: if  $r$  is not part of the input then
3:    $r = \lceil \sum_i x(i) \rceil$ ;
4: else
5:    $x \leftarrow r \frac{x}{\sum_i x(i)}$ ;
6: end if
7:  $\mathcal{K} = \mathcal{K}^* = \left\{ k \mid x(k) > \frac{r}{r+1} \right\}$  and  $\nu = \nu^* = \max(2\epsilon, \min_{\{(i,j) \mid D(i,j) > 0\}} D(i, j))$ ;
8: while  $|\mathcal{K}| < r$  and  $\nu < \max_{i,j} D(i, j)$  do
9:    $\mathcal{S}_i = \{j \mid D(i, j) \leq \nu\}$  for  $1 \leq i \leq n$ ;
10:   $w(i) = \sum_{j \in \mathcal{S}_i} x(j)$  for  $1 \leq i \leq n$ ;
11:   $\mathcal{K} = \emptyset$ ;
12:  while  $\max_{1 \leq i \leq n} w(i) > \frac{r}{r+1}$  do
13:     $k = \operatorname{argmax} w(i)$ ;  $\mathcal{K} \leftarrow \mathcal{K} \cup \{k\}$ ;
14:    For all  $1 \leq i \leq n$  and  $j \in \mathcal{S}_k \cup \mathcal{S}_i$  :  $w(i) \leftarrow w(i) - x(j)$ ;
15:  end while
16:  if  $|\mathcal{K}| > |\mathcal{K}^*|$  then
17:     $\mathcal{K}^* = \mathcal{K}$ ;  $\nu = \nu^*$ ;
18:  end if
19:   $\nu \leftarrow 2\nu$ ;
20: end while
21: % Safety procedure in case the conditions of Theorem 3 are not satisfied:
22: if  $|\mathcal{K}^*| < r$  then
23:   $d = \max_{i,j} D(i, j)$ ;
24:   $\mathcal{S}_i = \{j \mid D(i, j) \leq \nu^*\}$  for  $1 \leq i \leq n$ ;
25:   $w(i) = \sum_{j \in \mathcal{S}_i} x(j)$  for  $1 \leq i \leq n$ ;
26:   $\mathcal{K}^* = \emptyset$ ;
27:  while  $|\mathcal{K}^*| < r$  do
28:     $k = \operatorname{argmax} w(i)$ ;  $\mathcal{K}^* \leftarrow \mathcal{K}^* \cup \{k\}$ ;
29:    For all  $1 \leq i \leq n$ , and  $j \in \mathcal{S}_k \cup \mathcal{S}_i$  :  $w(i) \leftarrow w(i) - \left( \frac{d - D(i, j)}{d} \right)^{0.1} x(j)$ ;
30:     $w(k) \leftarrow 0$ ;
31:  end while
32: end if
```

Algorithm 5 Extracting Columns of a Noisy Separable Matrix with Outliers by Linear Programming

Input: A normalized noisy r -separable matrix $\tilde{M} = [W, T, WH']\Pi + N \in \mathbb{R}_+^{m \times n}$ with outliers, the noise level $\|N\|_1 \leq \epsilon$ and a vector $p \in \mathbb{R}^n$ with positive distinct entries and $\rho = 2$.

Output: An m -by- r matrix \tilde{W} such that $\tilde{W} \approx W$ (up to permutation).

- 1: Compute the optimal solution X^* of (3) where p has distinct positive entries.
 - 2: Let $\mathcal{K} = \{1 \leq k \leq n \mid X^*(k, k) \geq \frac{1}{2} \text{ and } \|X^*(k, :)\|_1 - X^*(k, k) \geq \frac{1}{2}\}$.
 - 3: $\tilde{W} = \tilde{M}(:, \mathcal{K})$.
-

In order for Algorithm 5 to extract the correct set of columns of \tilde{M} , the off-diagonal entries of the rows corresponding to the columns of T (resp. columns of W) must be small (resp. large). This can be guaranteed using the following conditions (see also Theorem 4 below):

- The angle between the cone generated by the columns of T and the columns space of W is positive. More precisely, we will assume that for all $1 \leq k \leq t$

$$\min_{\substack{x \in \mathbb{R}_+^t, x(k) = 1 \\ y \in \mathbb{R}^r}} \|Tx - Wy\|_1 \geq \eta > 0. \quad (5)$$

In fact, if a nonnegative linear combination of outliers (that is, Tx with $x \geq 0$) belongs to the column space of W , then some data points can usually be reconstructed using a non-zero weight for these outliers (it suffices that some data points belong to the convex hull of some columns of W and that linear combination of outliers).

- The matrix $[W, T]$ is robustly conical, otherwise some columns of T could be reconstructed using other columns of T whose corresponding rows could hence have large off-diagonal entries.
- Each column of W is necessary to reconstruct at least one data point, otherwise the off-diagonal entries of the row of X^* corresponding to that ‘useless’ column of W will be small, possibly equal to zero, and it cannot be distinguished from an outlier. More formally, for all $1 \leq k \leq r$, there is a least one data point $M(:, j) = WH(:, j) \neq W(:, k)$ such that

$$\min_{x \geq 0, y \geq 0} \|M(:, j) - Tx - W(:, \mathcal{K})y\|_1 \geq \delta, \quad \text{where } \mathcal{K} = \{1, 2, \dots, r\} \setminus \{k\}. \quad (6)$$

If Equation (5) holds, this condition is satisfied for example when $\text{conv}(W)$ is a simplex and some points lie inside that simplex (it is actually satisfied if and only if each column of W define with other columns of W a simplex containing at least one data point in its interior).

These conditions allow to distinguish the columns of W from the outliers using off-diagonal entries of an optimal solution X^* of (3):

Theorem 4. Suppose $\tilde{M} = M + N$ where the columns of M sum to one, $M = [W, T]H$ has the form (4) with $H \geq 0$, $\max_{ij} H'_{ij} \leq \beta \leq 1$ and $[W, T]$ κ -robustly conical, and $\|N\|_1 \leq \epsilon$. Suppose also that M , W and T satisfy Equations (5) and (6) for some $\eta > 0$ and $\delta > 0$. If

$$\epsilon \leq \frac{\nu(1 - \beta)}{20(n - 1)} \quad \text{where } \nu = \min(\kappa, \eta, \delta),$$

then Algorithm 5 extracts a matrix $\tilde{W} \in \mathbb{R}^{m \times r}$ satisfying $\|W - \tilde{W}(:, P)\|_1 \leq \epsilon$ for some permutation P .

Proof. See Appendix D. □

Unfortunately, the factor $\frac{1}{n-1}$ is necessary because a row of X^* corresponding to an outlier could potentially be assigned weights proportional to ϵ for all off-diagonal entries. For example, if all data points are perturbed in the direction of an outlier, that is, $N(:, j) = \epsilon T(:, k)$ for all j and for some $1 \leq k \leq t$, then we could have $\sum_{j \neq k} X(k, j) = (n - 1)\mathcal{O}(\epsilon)$ hence it is necessary that $\epsilon \leq \mathcal{O}(n^{-1})$ (although it is not likely to happen in practice). A simple way to improve the bound is the following:

- Identify the vertices and outliers using $\mathcal{K} = \{1 \leq k \leq n \mid X^*(k, k) \geq \frac{1}{2}\}$ (this only requires $\epsilon \leq \frac{\kappa(1-\beta)}{20}$, cf. Theorem 1).
- Solve the linear program $Z^* = \text{argmin}_{Z \geq 0} \|M - M(:, \mathcal{K})Z\|_1$.

- Use the sum of the rows of Z^* (instead of X^*) to identify the columns of W .

Following the same steps as in the proof of Theorem 4, the bound for ϵ for the corresponding algorithm becomes $\epsilon \leq \frac{\nu(1-\beta)}{20(r+t-1)}$.

Remark 6 (Number of outliers). *Algorithm 5 does not require the number of outliers as an input. Moreover, the number of outliers is not limited hence our result is stronger than in [11] where the number of outliers cannot exceed $m - r$ (because T needs to be full rank, while we only need T to be robustly conical and the cone generated by its columns define a wide angle with the column space of W).*

Remark 7 (Hottopixx and outliers). *Replacing the constraint $\text{tr}(X) = r$ with $\text{tr}(X) = r + t$ (r is the number of columns of W and t is the number of outliers) in the LP model (2) allows to deal with outliers. However, the number of outliers plus the number of columns of W (that is, $r + t$) has to be estimated, which is rather impractical.*

4 Avoiding Column Normalization

In order to use the LP models (2) and (3), normalization must be enforced which may introduce significant distortions in the dataset and lead to poor performances [13]. If M is r -separable but its columns do not sum to one, we still have that

$$M = W[I_r, H']\Pi = [W, WH']\Pi = [W, WH'] \begin{pmatrix} I_r & H' \\ 0_{(n-r) \times r} & 0_{(n-r) \times (n-r)} \end{pmatrix} \Pi = MX^0.$$

However, the constraints $X(i, j) \leq X(i, i)$ for all i, j in the LP's (2) and (3) are not necessarily satisfied by the matrix X^0 , because the entries of H' can be arbitrarily large.

Let us denote \tilde{M}_o the original unnormalized noisy data matrix, and its normalized version \tilde{M} , with

$$\tilde{M}(:, j) = \frac{\tilde{M}_o(:, j)}{\|\tilde{M}_o(:, j)\|_1} \quad \text{for all } j.$$

Let us also rewrite the LP (3) in terms of \tilde{M}_o instead of \tilde{M} using the following change of variables

$$X_{ij} = \frac{\|\tilde{M}_o(:, i)\|_1}{\|\tilde{M}_o(:, j)\|_1} Y_{ij} \quad \text{for all } i, j. \quad (7)$$

Note that $Y_{ii} = X_{ii}$ for all i . We have for all j that

$$\begin{aligned} \left\| \tilde{M}(:, j) - \sum_i \tilde{M}(:, i) X_{ij} \right\|_1 &= \left\| \frac{\tilde{M}_o(:, j)}{\|\tilde{M}_o(:, j)\|_1} - \sum_j \frac{\tilde{M}_o(:, i)}{\|\tilde{M}_o(:, i)\|_1} \frac{\|\tilde{M}_o(:, i)\|_1}{\|\tilde{M}_o(:, j)\|_1} Y_{ij} \right\|_1 \\ &= \frac{1}{\|\tilde{M}_o(:, j)\|_1} \left\| \tilde{M}_o(:, j) - \sum_j \tilde{M}_o(:, i) Y_{ij} \right\|_1, \end{aligned}$$

which proves that the following LP

$$\min_{Y \in \mathcal{Y}} p^T \text{diag}(Y) \quad \text{such that} \quad \|\tilde{M}_o(:, j) - \tilde{M}_o Y(:, j)\|_1 \leq \rho \epsilon \|\tilde{M}_o(:, j)\|_1 \quad \text{for all } j, \quad (8)$$

where

$$\mathcal{Y} = \{Y \in \mathbb{R}_+^{n \times n} \mid Y(i, i) \leq 1 \text{ for all } i, \text{ and } \|\tilde{M}_o(:, i)\|_1 Y(i, j) \leq \|\tilde{M}_o(:, j)\|_1 Y(i, i) \text{ for all } i, j\}, \quad (9)$$

is equivalent to the LP (3). This shows that the LP (3) looks for an approximation $\tilde{M}_o Y$ of \tilde{M}_o with small *relative error*, which is in general not desirable in practice. For example, a zero column to which some noise is added will have to be approximated rather well, while it does not bring any valuable information. Similarly, the columns of M with large norms will be given relatively less importance while they typically contain a more reliable information (e.g., in document datasets, they correspond to longer documents).

It is now easy to modify the LP (8) to handle other noise models. For example, if the noise added to each column of the input data matrix is independent of its norm, then one should rather use the following LP trying to find an approximation $\tilde{M}_o Y$ of \tilde{M}_o with small *absolute error*:

$$\min_{Y \in \mathcal{Y}} p^T \text{diag}(Y) \quad \text{such that} \quad \|\tilde{M}_o - \tilde{M}_o Y\|_1 \leq \rho\epsilon. \quad (10)$$

Remark 8 (Other noise models). *Considering other models depending is also possible: one has to replace the constraint $\|\tilde{M}_o - \tilde{M}_o Y\|_1 \leq \rho\epsilon$ with another appropriate constraint. For example, using any ℓ_q -norm with $q \geq 1$ leads to efficiently solvable convex optimization programs [12], that is, using*

$$\|\tilde{M}_o(:, j) - \tilde{M}_o Y(:, j)\|_q \leq \rho\epsilon, \quad \text{for all } j.$$

Another possibility is to assume that the noise is distributed among all the entries of the input matrix independently (that is, some columns could have been contaminated with more noise than others) and one could use instead $\sqrt[q]{\sum_{i,j} (\tilde{M}_o - \tilde{M}_o Y)_{ij}^q} \leq \rho\epsilon$, e.g., $\|\tilde{M}_o - \tilde{M}_o Y\|_F \leq \rho\epsilon$ for Gaussian noise (where $\|\cdot\|_F$ is the Frobenius norm of a matrix with $q = 2$).

5 Numerical Experiments

In this section, we present some numerical experiments in which we compare our new LP model (10) with Hottopixx and two other state-of-the-art methods. First we describe a practical twist to Algorithm 4, which we routinely apply in the experiments to LP-based solutions.

5.1 Post-Processing of LP solutions

Recall that the LP-based algorithms return a nonnegative matrix X whose diagonal entries indicate the importance of the corresponding columns of the input data matrix \tilde{M} . As explained earlier, there are several ways to extract r columns from \tilde{M} using this information, the simplest being to select the columns corresponding to the r largest diagonal entries of X [4]. Another approach is to take into account the distances between the columns of \tilde{M} and cluster them accordingly; see Algorithm 4. In our experiments we have not observed that one method dominates the other (although in theory, when the noise level is sufficiently small, Algorithm 4 is more robust [9]). Therefore, the strategy we employ in the experiments below selects the best solution out of the two post-processing strategies based on the residual error, see Algorithm 6.

5.2 Algorithms

In this section, we compare the following near-separable NMF algorithms:

1. **Hottopixx** [4]. Given the noise level $\|N\|_1$ and the factorization rank r , it computes the optimal solution X^* of the LP (2) (where the input matrix \tilde{M} has to be normalized) and returns the indices obtained using Algorithm 6. The vector p in the objective function was randomly generated using the `randn` function of Matlab. The algorithm of Arora et al. [2] was shown to perform worse than Hottopixx [4] hence we do not include it here (moreover, it requires an additional parameter α related to the conditioning of W which is difficult to estimate in practice).

Algorithm 6 Hybrid Post-Processing for LP-based Near-Separable NMF Algorithms

Input: A matrix $M \in \mathbb{R}^{m \times n}$, a factorization rank r , a noise level ϵ , and a vector of weight $x \in \mathbb{R}_+^n$.

Output: An index set \mathcal{K} such that $\min_{H \geq 0} \|M - M(:, \mathcal{K})H\|_F$ is small.

- 1: % Greedy approach
 - 2: \mathcal{K}_1 is the set of the r largest indices of x ;
 - 3: % Clustering using Algorithm 4
 - 4: $\mathcal{K}_2 = \text{Algorithm 4}(\tilde{M}, x, \epsilon, r)$;
 - 5: % Select the better of the two
 - 6: $\mathcal{K} = \operatorname{argmin}_{\mathcal{R} \in \{\mathcal{K}_1, \mathcal{K}_2\}} \min_{H \geq 0} \|M - M(:, \mathcal{R})H\|_F^2$;
-

2. **SPA** [1]. The successive projection algorithm (SPA) extracts recursively r columns of the input normalized matrix \tilde{M} as follows: at each step, it selects the column with maximum ℓ_2 norm, and then projects all the columns of \tilde{M} on the orthogonal complement of the extracted column. This algorithm was proved to be robust to noise [11]. (Note that there exist variants where, at each step, the column is selected according to other criteria, e.g., any ℓ_p norm with $1 < p < +\infty$. This particular version of the algorithm using ℓ_2 norm actually dates back from modified Gram-Schmidt with column pivoting, see [11] and the references therein.) SPA was shown to perform significantly better on several synthetic datasets than Hottopixx and several state-of-the-art algorithms from the hyperspectral image community [11] (these algorithms are based on the pure-pixel assumption which is equivalent to the separability assumption, see Introduction).
3. **XRAY** [13]. In [13], several fast conical hull algorithms are proposed. We use in this paper the variant referred to as *max*, because it performs in average the best on synthetic datasets. Similarly as SPA, it recursively extracts r columns of the input unnormalized matrix \tilde{M}_o : at each step, it selects a column of \tilde{M}_o corresponding to an extreme ray of the cone generated by the columns of \tilde{M}_o , and then projects all the columns of \tilde{M}_o on the cone generated by the columns of \tilde{M}_o extracted so far. XRAY was shown to perform much better than Hottopixx and similarly as SPA on synthetic datasets (while performing better than both for topic identification in document datasets as it does not require column normalization). However, it is not known whether XRAY is robust to noise.
4. **LP** (10) with $\rho = 1, 2$. Given the noise level $\|N\|_1$, it computes the optimal solution X^* of the LP (10) and returns the indices obtained with the post-processing described in Algorithm 6. (Note that we have also tried $\rho = \frac{1}{2}$ which performs better than $\rho = 2$ but slightly worse than $\rho = 1$ in average hence we do not display these results here.)

Table 1 gives the following information for the different algorithms: computational cost, memory requirement, parameters and if column normalization of the input matrix is necessary.

	Flops	Memory	Parameters	Normalization
Hottopixx [4]	$\Omega(mn^2)$	$\mathcal{O}(mn + n^2)$	$\ N\ _1, r$	Yes
SPA [1, 11]	$2mnr + \mathcal{O}(mr^2)$	$\mathcal{O}(mn)$	r	Yes
XRAY [13]	$\mathcal{O}(mnr)$	$\mathcal{O}(mn)$	r	No
LP (10)	$\Omega(mn^2)$	$\mathcal{O}(mn + n^2)$	$\ N\ _1$	No

Table 1: Comparison of robust algorithms for near-separable NMF for a dense m -by- n input matrix.

The LP have been solved using the IBM ILOG CPLEX Optimizer² on a standard Linux box. Because of the greater complexity of the LP-based approaches (formulating (2) and (10) as LP's requires $n^2 + mn$ variables), the size of the input data matrices allowed on a standard machine is limited, roughly $mn^2 \sim 10^6$ (for example, on a two-core machine with 2.99GHz and 2GB of RAM, it already takes about one minute to process a 100-by-100 matrix using CPLEX). In this paper, we focus on the robustness performance of the different algorithms and only compare them on synthetic datasets. Comparison on large-scale real-world datasets would require dedicated implementations, such as the parallel first-order method proposed in [4] for the LP (2), and is a topic for further research. The code for all algorithms is available at <https://sites.google.com/site/nicolasgillis/code>.

5.3 Synthetic Datasets

With the algorithms above we have run a benchmark with certain synthetic datasets particularly suited to investigate the robustness behaviour under influence of noise. In all experiments the problem dimensions are fixed to $m = 50$, $n = 100$ and $r = 10$. We conducted our experiments with six different data models. As we will describe next, the models differ in the way the factor H is constructed and the sparsity of the noise matrix N . Given a desired noise level ϵ , the noisy r -separable matrix $\tilde{M} = M + N = WH + N$ is generated as follows:

The entries of W are drawn uniformly at random from the interval $[0, 1]$ (using Matlab's `rand` function). Then the columns of W are normalized to sum to one.

The first r columns of H are always taken as the identity matrix to satisfy the separability assumption. The remaining columns of H and the noise matrix N are generated in two different ways (similar to [11]):

1. *Dirichlet*. The remaining 90 columns of H are generated according to a Dirichlet distribution whose r parameters are chosen uniformly in $[0, 1]$ (the Dirichlet distribution generates vectors on the boundary of the unit simplex so that $\|H(:, j)\|_1 = 1$ for all j). Each entry of the noise matrix N is generated following the normal distribution $\mathcal{N}(0, 1)$ (using the `randn` function of Matlab).
2. *Middle Points*. The $\frac{r(r-1)}{2} = 45$ next columns of H resemble all possible equally weighted convex combinations of pairs from the r leading columns of H . This means that the corresponding 45 columns of M are the middle points of pairs of columns of W . The trailing 45 columns of H are generated in the same way as above, using the Dirichlet distribution. No noise is added to the first r columns of M , that is, $N(:, 1 : r) = 0$, while all the other columns are moved toward the exterior of the convex hull of the columns of W using

$$N(:, j) = M(:, j) - \bar{w}, \quad \text{for } r + 1 \leq j \leq n,$$

where \bar{w} is the average of the columns of W (geometrically, this is the vertex centroid of the convex hull of the columns of W).

We combine these two choices for H and N with three options that control the pattern density of N , thus yielding a total of six different data models:

1. *Dense noise*. Leave the matrix N untouched.
2. *Sparse noise*. Apply a mask to N such that roughly 75% of the entries are set to zero (using the `density` parameter of Matlab's `sprand` function).
3. *Pointwise noise*. Keep only one randomly picked non-zero entry in each nonzero column of N .

Finally we scale the resulting matrix N by a scalar such that $\|N\|_1 = \epsilon$. In order to avoid a bias towards the natural ordering, the columns of \tilde{M} are permuted at random in a last step.

²Available for free at <http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/> for academia.

5.4 Error Measures and Methodology

Let \mathcal{K} be the set of indices extracted by an algorithm. In our comparisons, we will use the following two error measures:

- *Index recovery*: percentage of correctly extracted indices in \mathcal{K} (recall that we *know* the indices corresponding to the columns of W).
- ℓ_1 *residual norm*: Denote by $\|A\|_s = \sum_{ij} |a_{ij}|$ the ℓ_1 norm for matrices. We measure the relative ℓ_1 residual by

$$1 - \min_{H \geq 0} \frac{\|\tilde{M} - \tilde{M}(:, \mathcal{K})H\|_s}{\|\tilde{M}\|_s}.$$

Note that both measures are between zero and one, one being the best possible value, zero the worst.

The aim of the experiments is to display the robustness of the algorithms from Section 5.2 applied to the data sets described in the previous section under increasing noise levels. For each data model, we ran all the algorithms on the same randomly generated data on a predefined range of noise levels ϵ . For each such noise level, 25 data sets were generated and the two measures are averaged over this sample for each algorithm.

5.5 Results

Figures 1 and 2 display the results for the three experiments of “Dirichlet” and “Middle Points” types respectively. In all experiments, we observe that

- The new LP model (10) is significantly more robust to noise than Hottopixx, which confirms our theoretical results; see Section 2.1.
- The variant of LP (10) with $\rho = 2$ is less robust than with $\rho = 1$, as suggested by our theoretical findings from Section 2.1.
- SPA and XRAY perform, in average, very similarly.

Comparing the three best algorithms (that is, SPA, XRAY and LP (10) with $\rho = 1$), we have that

- In case of “dense” noise, they give comparable results; although LP (10) with $\rho = 1$ performs slightly worse for the “Dirichlet” type, and slightly better for the “Middle Points” type.
- In case of “sparse” noise, LP (10) with $\rho = 1$ performs consistently better than SPA and XRAY: for all noise levels, it identifies correctly more columns of W and the corresponding NMF’s have smaller ℓ_1 residual norms.
- In case of “pointwise” noise, LP (10) with $\rho = 1$ outperforms SPA and XRAY. In particular, for high noise level, it is able to extract correctly almost all columns of W while SPA and XRAY can only extract a few for the “Dirichlet” type (performing as a guessing algorithm since they extract correctly only $r/n = 10\%$ of the columns of W), or none for the “Middle Points” type.

Note that LP (10) with $\rho = 2$ also performs consistently better than SPA and XRAY in case of “pointwise” noise.

Remark 9. For the “Middle Points” experiments and for large noise levels, the middle points of the columns of W become the vertices of the convex hull of the columns of \tilde{M} (since they are perturbed toward the outside of the convex hull of the columns of W). Hence, near-separable NMF algorithms should not extract any original column of W . However, the index measure for LP (10) with $\rho = 2$ increases for larger noise level (although the ℓ_1 residual measure decreases); see Figure 2. It is difficult to explain this behavior because the noise level is very high (close to 100%) hence the separability assumption is far from being satisfied and it is not clear what the LP (10) does.

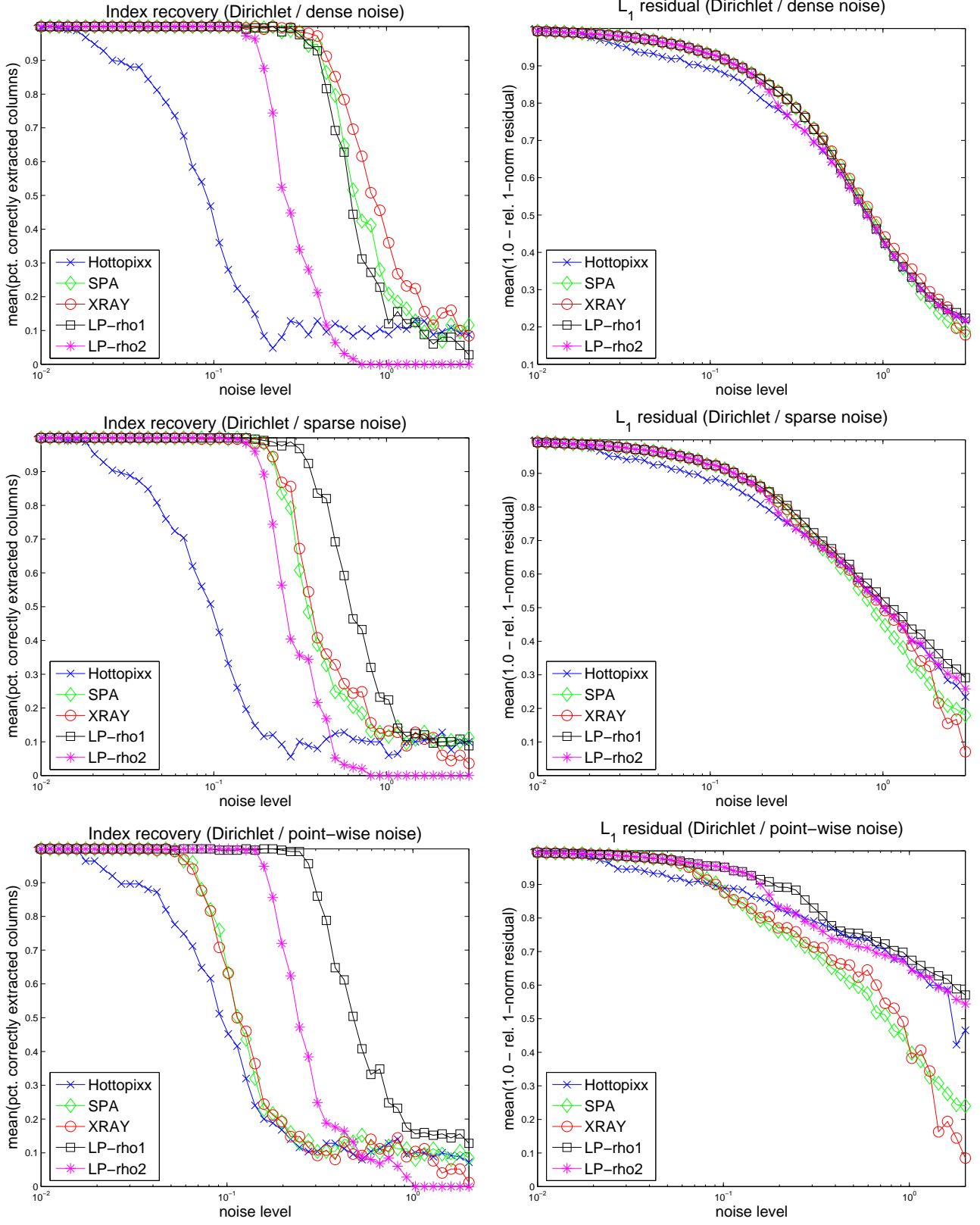


Figure 1: Comparison of near-separable NMF algorithms on “Dirichlet” type data sets. From left to right: index recovery and ℓ_1 residual. From top to bottom: dense noise, sparse noise and pointwise noise.

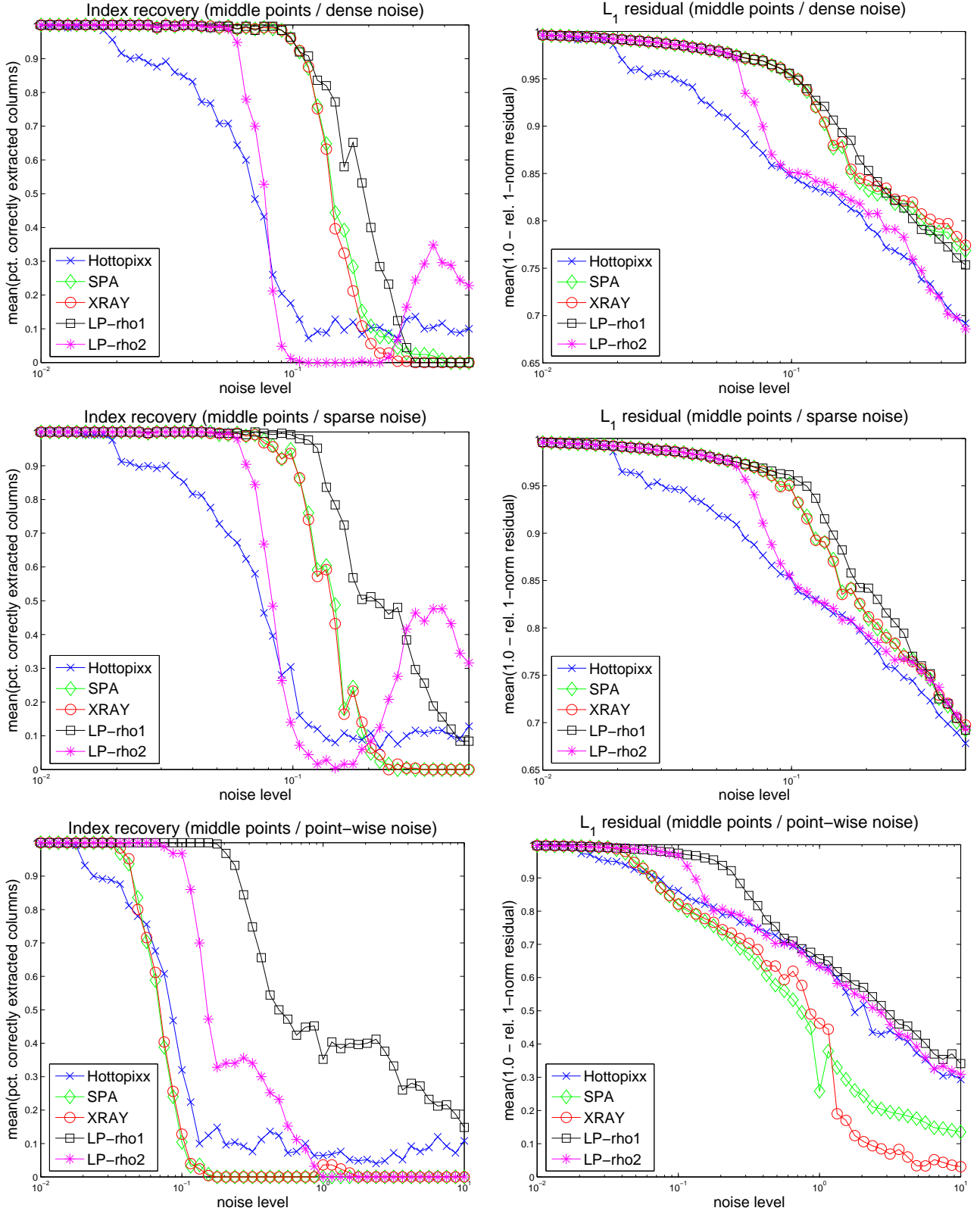


Figure 2: Comparison of near-separable NMF algorithms on “Middle Points” type data sets. From left to right: index recovery and ℓ_1 residual. From top to bottom: dense noise, sparse noise and pointwise noise.

	D/dense	D/sparse	D/pw	MP/dense	MP/sparse	MP/pw
Hottopixx	2.5	2.5	3.6	4.4	4.3	4.2
SPA	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
XRAY	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
LP (10), $\rho = 1$	20.5	34.1	39.0	52.5	88.1	41.4
LP (10), $\rho = 2$	10.5	12.3	16.0	32.5	56.9	27.4

Table 2: Average computational time in seconds for the different algorithms and data models. (D stands for Dirichlet, MP for middle points, pw for pointwise.)

	D/dense	D/sparse	D/pw	MP/dense	MP/sparse	MP/pw
Hottopixx	0.014	0.018	0.016	0.016	0.018	0.015
SPA	0.220	0.154	0.052	0.077	0.071	0.032
XRAY	0.279	0.154	0.052	0.083	0.071	0.032
LP (10), $\rho = 1$	0.279	0.195	0.197	0.083	0.098	0.178
LP (10), $\rho = 2$	0.137	0.121	0.141	0.055	0.055	0.075

Table 3: Index recovery robustness: Largest noise level $\|N\|_1$ for which an algorithm achieves almost perfect index recovery (that is, at least 99% on average).

Table 2 gives the average computational time for a single application of the algorithms to a dataset. As expected, the LP-based methods are significantly slower than SPA and XRAY; designing faster solvers is definitely an important topic for further research. Note that the Hottopixx model can be solved about ten times faster on average than the LP model (10), despite the only essential difference being the trace constraint $\text{tr}(X) = r$. It is difficult to explain this behaviour as the number of simplex iterations or geometry of the central path cannot easily be set in relation to the presence or absence of a particular constraint.

Table 3 displays the index recovery robustness: For each algorithm and data model, the maximum noise level $\|N\|_1$ for which the algorithm recovered on average at least 99% of the indices corresponding to the columns of W . In all cases, the LP (10) with $\rho = 1$ is on par or better than all other algorithms.

6 Conclusion and Further Work

In this paper, we have proposed a new more practical and more robust LP model for near-separable NMF which competes favorably with two state-of-the-art methods (outperforming them in some cases). It would be particularly interesting to investigate the following directions of research:

- Implementation and evaluation of an algorithm to solve (10) for large-scale real-world problems.
- Improvement of the theoretical bound on the noise level for Algorithm 3 to extract the right set of columns of the input data matrix in case duplicates and near duplicates are present in the dataset (cf. Section 2.2).
- Design of practical and robust near-separable NMF algorithms. For example, would it be possible to design an algorithm as robust as our LP-based approach but computationally more effective (e.g., running in $\mathcal{O}(mnr)$ operations)?

A Proof of Theorem 1

The next two lemmas are simple generalizations of [9, Lemmas 2 & 3].

Lemma 1. Suppose $\tilde{M} = M + N$ where $\|M(:, j)\|_1 = 1$ for all j and $\|N\|_1 \leq \epsilon < 1$, and suppose X is a feasible solution of (3). Then,

$$\|X\|_1 \leq 1 + \epsilon \left(\frac{\rho + 2}{1 - \epsilon} \right) \quad \text{and} \quad \|M - MX\|_1 \leq \epsilon \left(\frac{\rho + 2}{1 - \epsilon} \right).$$

Proof. First note that $\|\tilde{M}\|_1 = \|M + N\|_1 \leq \|M\|_1 + \|N\|_1 \leq 1 + \epsilon$. By the feasibility of X for (3),

$$\rho\epsilon \geq \|\tilde{M} - \tilde{M}X\|_1 \geq \|\tilde{M}X\|_1 - \|\tilde{M}\|_1 \geq \|MX\|_1 - \|NX\|_1 - (1 + \epsilon) \geq \|X\|_1 - \epsilon\|X\|_1 - 1 - \epsilon,$$

hence $\|X\|_1 \leq 1 + \epsilon \left(\frac{\rho + 2}{1 - \epsilon} \right)$, implying that $\|NX\|_1 \leq \|N\|_1 \|X\|_1 \leq \epsilon \left(1 + \frac{(\rho + 2)\epsilon}{1 - \epsilon} \right)$. Therefore

$$\rho\epsilon \geq \|\tilde{M} - \tilde{M}X\|_1 = \|M + N - (M + N)X\|_1 \geq \|M - MX\|_1 - \epsilon - \epsilon \left(1 + \frac{(\rho + 2)\epsilon}{1 - \epsilon} \right),$$

from which we obtain $\|M - MX\|_1 \leq \epsilon \left(\rho + 2 + \frac{(\rho + 2)\epsilon}{1 - \epsilon} \right) = \epsilon \left(\frac{\rho + 2}{1 - \epsilon} \right)$ \square

Lemma 2. Let $\tilde{M} = M + N$ where $\|M(:, j)\|_1 = 1$ for all j , admits a rank- r separable factorization WH with W κ -robustly conical and $\|N\|_1 \leq \epsilon < 1$, and has the form (1) with $\max_{i,j} H'_{ij} \leq \beta < 1$ and $W, H \geq 0$. Let also X be any feasible solution of (3), then

$$X(j, j) \geq 1 - \frac{2\epsilon}{\kappa(1 - \beta)} \left(\frac{\rho + 2}{1 - \epsilon} \right) \quad \text{for all } j \text{ such that } M(:, j) = W(:, k) \text{ for some } 1 \leq k \leq r.$$

Proof. Let \mathcal{K} be the set of r indices such that $M(:, \mathcal{K}) = W$. Let also $1 \leq k \leq r$ and denote $j = \mathcal{K}(k)$ so that $M(:, j) = W(:, k)$. By Lemma 1,

$$\|W(:, k) - WHX(:, j)\|_1 \leq \epsilon \left(\frac{\rho + 2}{1 - \epsilon} \right). \quad (11)$$

Since $H(k, j) = 1$,

$$\begin{aligned} WHX(:, j) &= W(:, k)H(k, :)X(:, j) + W(:, \mathcal{R})H(\mathcal{R}, :)X(:, j) \\ &= W(:, k) \left(X(j, j) + H(k, \mathcal{J})X(\mathcal{J}, j) \right) + W(:, \mathcal{R})y, \end{aligned}$$

where $\mathcal{R} = \{1, 2, \dots, r\} \setminus \{k\}$, $\mathcal{J} = \{1, 2, \dots, n\} \setminus \{j\}$ and $y = H(\mathcal{R}, :)X(:, j) \geq 0$. We have

$$\eta = X(j, j) + H(k, \mathcal{J})X(\mathcal{J}, j) \leq X(j, j) + \beta \left(1 + \frac{(\rho + 2)\epsilon}{1 - \epsilon} - X(j, j) \right), \quad (12)$$

since $\|H(k, \mathcal{J})\|_\infty \leq \beta$ and $\|X(:, j)\|_1 \leq 1 + \frac{(\rho + 2)\epsilon}{1 - \epsilon}$ (Lemma 1). Hence

$$\|W(:, k) - WHX(:, j)\|_1 \geq (1 - \eta) \left\| W(:, k) - W(:, \mathcal{R}) \frac{y}{1 - \eta} \right\|_1 \geq (1 - \eta)\kappa. \quad (13)$$

Combining Equations (11), (12) and (13), we obtain

$$1 - \left(X(j, j) + \beta \left(1 + \frac{(\rho + 2)\epsilon}{1 - \epsilon} - X(j, j) \right) \right) \leq \frac{\epsilon}{\kappa} \left(\frac{\rho + 2}{1 - \epsilon} \right)$$

which gives, using the fact that $\kappa, \beta \leq 1$,

$$X(j, j) \geq 1 - \frac{2\epsilon}{\kappa(1 - \beta)} \left(\frac{\rho + 2}{1 - \epsilon} \right). \quad \square$$

Lemma 3. Let $\tilde{M} = M + N$ where $\|M(:, j)\|_1 = 1$ for all j , admits a rank- r separable factorization WH and $\|N\|_1 \leq \epsilon$, and has the form (1). Let \mathcal{K} be the index set with r elements such that $M(:, \mathcal{K}) = W$. Let also X^* be an optimal solution of (3) such that

$$X^*(k, k) \geq \gamma \quad \text{for all } k \in \mathcal{K}, \quad (14)$$

where $0 \leq \gamma \leq 1$. Then,

$$X^*(j, j) \leq 1 - \min\left(\gamma, \frac{\rho}{2}\right) \quad \text{for all } j \notin \mathcal{K}.$$

Proof. Let X be any feasible solution of (3) satisfying (14), and $\alpha = \min(\gamma, \frac{\rho}{2})$. Let us show that the j th column of X for some $j \notin \mathcal{K}$ can be modified as follows

$$X(i, j) \leftarrow \begin{cases} 1 - \alpha & \text{if } i = j, \\ \alpha H(i, j) & \text{if } i \in \mathcal{K}, \\ 0 & \text{otherwise,} \end{cases}$$

while keeping feasibility. First, $\alpha H(i, j) \leq \gamma \leq X(i, i)$ for all $i \in \mathcal{K}$ hence the condition $X(i, j) \leq X(i, i)$ for all i, j is satisfied while, clearly, $0 \leq X(i, i) \leq 1$ for all i . It remains to show that $\|\tilde{M}(:, j) - \tilde{M}X(:, j)\|_1 \leq \rho\epsilon$. By assumption, $M(:, j) = WH(:, j) = \alpha WH(:, j) + (1 - \alpha)M(:, j)$ hence

$$\begin{aligned} \tilde{M}(:, j) &= \alpha (M(:, j) + N(:, j)) + (1 - \alpha)\tilde{M}(:, j) \\ &= \alpha (WH(:, j) + N(:, j)) + (1 - \alpha)\tilde{M}(:, j). \end{aligned}$$

This gives

$$\|\tilde{M}(:, j) - \tilde{M}X(:, j)\|_1 = \alpha \|M(:, j) + N(:, j) - (W + N(:, \mathcal{K}))H(:, j)\|_1 \leq 2\alpha\epsilon \leq \rho\epsilon,$$

since the columns of H sum to one, and $\|N\|_1 \leq \epsilon$. This result implies that any optimal solution X^* satisfying (14) must satisfy $X^*(j, j) \leq 1 - \alpha$, otherwise we could replace the j th column of X^* using the construction above and obtain a strictly better solution since the vector p in the objective function only has positive entries. \square

Proof of Theorem 1. Let X be an optimal solution of (3). Let us first consider the case $\epsilon = 0$, which is particular because it allows duplicates of the columns of W in the dataset and the value of ρ does not influence the analysis since $\rho\epsilon = 0$ for any $\rho > 0$. Let denote

$$\mathcal{K}_k = \{j \mid M(:, j) = W(:, k)\},$$

the set of indices whose corresponding column of M is equal to the k th column of W . By assumption, $\kappa > 0$ hence for all $1 \leq k \leq r$ we have $W(:, k) \notin \text{cone}(W(:, \bar{\mathcal{K}}))$ where $\bar{\mathcal{K}} = \{1, 2, \dots, r\} \setminus \{k\}$. This implies that $\sum_{j \in \mathcal{K}_k} X(j, j) \geq 1$ for all k . Since we are minimizing a positive linear combination of the diagonal entries of X and assigning a weight of one to each cluster \mathcal{K}_k is feasible (see Equation 1), we have $\sum_{j \in \mathcal{K}_k} X(j, j) = 1$. Moreover, assigning all the weight to the index in \mathcal{K}_k with the smallest entry in p minimizes the objective function (and this index is unique since the entries of p are distinct). Finally, for all $1 \leq k \leq r$, there exists a unique j such that $M(:, j) = W(:, k)$ and $X(j, j) = 1$ which gives the result for $\epsilon = 0$.

Otherwise $\epsilon > 0$ and $\beta < 1$, and the result follows from Lemmas 2 and 3: Let \mathcal{K} be the set of r indices such that $M(:, \mathcal{K}) = W$. By Lemma 2, we have

$$X(k, k) \geq 1 - \frac{2\epsilon}{\kappa(1 - \beta)} \left(\frac{\rho + 2}{1 - \epsilon} \right), \quad \text{for all } k \in \mathcal{K},$$

while, by Lemma 3,

$$X(j, j) \leq \max\left(1 - \frac{\rho}{2}, \frac{2\epsilon}{\kappa(1 - \beta)} \left(\frac{\rho + 2}{1 - \epsilon} \right)\right), \quad \text{for all } j \notin \mathcal{K}.$$

Therefore, if

$$1 - \frac{2\epsilon}{\kappa(1-\beta)} \left(\frac{\rho+2}{1-\epsilon} \right) > f \geq \max \left(1 - \frac{\rho}{2}, \frac{2\epsilon}{\kappa(1-\beta)} \left(\frac{\rho+2}{1-\epsilon} \right) \right),$$

where $f = 1 - \frac{\min(1, \rho)}{2} = \max(\frac{1}{2}, 1 - \frac{\rho}{2})$, then Algorithm 2 extracts the r indices corresponding to the columns of W . The above conditions are satisfied if

$$\frac{2\epsilon}{\kappa(1-\beta)} \left(\frac{\rho+2}{1-\epsilon} \right) < \frac{\rho}{2} \quad \text{and} \quad \frac{2\epsilon}{\kappa(1-\beta)} \left(\frac{\rho+2}{1-\epsilon} \right) < \frac{1}{2},$$

that is, $\frac{\epsilon}{1-\epsilon} < \frac{\kappa(1-\beta) \min(1, \rho)}{4(\rho+2)}$. Taking

$$\epsilon \leq \frac{\kappa(1-\beta) \min(1, \rho)}{5(\rho+2)} < \frac{\kappa(1-\beta) \min(1, \rho)}{\rho+2} \frac{1-\epsilon}{4}$$

gives the results since $\epsilon \leq \frac{1}{5(\rho+2)} < \frac{1}{10}$ for any $\rho > 0$ hence $\frac{1-\epsilon}{4} > \frac{1}{5}$. \square

B Proof of Theorem 2

Proof of Theorem 2. Let us consider

$$W = \begin{pmatrix} \frac{\kappa}{2} I_r \\ (1 - \frac{\kappa}{2}) e^T \end{pmatrix}, H = \begin{pmatrix} I_r & \beta I_r + \frac{1-\beta}{r-1} (E_r - I_r) \end{pmatrix}, \text{ and } N = 0,$$

where $\frac{1}{r} \leq \beta < 1$ and W is κ -robustly conical with $\kappa > 0$ [9]. Let also $p = \binom{K}{e}$ where e is the vector of all ones and K is a large constant. The matrix

$$X = \begin{pmatrix} \left(1 - \frac{\rho\epsilon}{\kappa(1-\beta)}\right) I_r & 0 \\ \frac{\rho\epsilon}{\kappa(1-\beta)} I_r & I_r \end{pmatrix}$$

is a feasible solution of (3) for any $\epsilon \leq \frac{\kappa(1-\beta)}{\rho}$. In fact, for all $1 \leq j \leq r$,

$$\|M(:, j) - MX(:, j)\|_1 = \frac{\rho\epsilon}{\kappa(1-\beta)} \|M(:, j) - M(:, j+r)\|_1 = \rho\epsilon,$$

while it can be easily checked that X satisfies the other constraints. By [9, Lemma 7], for K sufficiently large, any optimal solution X^* of (3) must satisfy

$$\min_{1 \leq k \leq r} X^*(k, k) \leq \max_{1 \leq k \leq r} X(k, k) = 1 - \frac{\rho\epsilon}{\kappa(1-\beta)},$$

(otherwise $p^T \text{diag}(X^*) > p^T \text{diag}(X)$ for K sufficiently large). For the columns of W to be extracted, one requires $X^*(k, k) > 1 - \frac{\min(1, \rho)}{2}$ for all $1 \leq k \leq r$ hence it is necessary that

$$1 - \frac{\rho\epsilon}{\kappa(1-\beta)} > 1 - \frac{\min(1, \rho)}{2} \iff \epsilon < \frac{\kappa(1-\beta)}{2} \frac{\min(1, \rho)}{\rho},$$

for Algorithm 2 to extract the right columns of M (that is, the first r ones). \square

C Proof of Theorem 3

Proof of Theorem 3. The matrix X^0 from Equation (1) is a feasible solution of (3); in fact,

$$\|\tilde{M} - \tilde{M}X^0\|_1 = \|M + N - (M + N)X^0\|_1 \leq \|M - MX^0\|_1 + \|N\|_1 + \|NX^0\|_1 \leq 2\epsilon,$$

since $M = MX^0$, $\|N\|_1 \leq \epsilon$ and $\|NX^0\|_1 \leq \|N\|_1\|X^0\|_1 \leq \epsilon$ as $\|X^0\|_1 = 1$. Therefore, since $p = e$, any optimal solution X^* of (3) satisfies

$$\text{tr}(X^*) = p^T \text{diag}(X^*) \leq p^T \text{diag}(X^0) = r.$$

The result then directly follows from [9, Theorem 5]. In fact, Algorithm 3 is exactly the same as [9, Algorithm 3] except that the optimal solution of (3) is used instead of (2) while [9, Theorem 5] does not need the entries of p to be distinct and only the condition $\text{tr}(X) \leq r$ is necessary. Note that [9, Theorem 5] guarantees that there are r disjoint clusters of columns of \tilde{M} around each column of W whose weight is strictly larger $\frac{r}{r+1}$. Therefore, the total weight is strictly larger than $r - \frac{r}{r+1} > r - 1$ while it is at most r (since $\text{tr}(X^*) \leq r$) implying that $r = \left\lceil \sum_{i=1}^n X^*(i, i) \right\rceil$. \square

D Proof of Theorem 4

Proof of Theorem 4. In case $\beta = 1$, $\epsilon = 0$ and the proof is similar to that of Theorem 1; the only difference is that the condition from Equation (5) has to be used to show that no weight can be assigned to off-diagonal entries of the rows of an optimal solution of (3) corresponding to the columns of T . Otherwise $\beta < 1$ and there are no duplicate nor near duplicate of the columns of W in the dataset.

Let assume without loss of generality that \tilde{M} has the form

$$\tilde{M} = [T, W, WH'] + N,$$

that is, the first t columns correspond to T and the r next ones to W . Let then X be an optimal solution of (3).

Since $[W, T]$ is κ -robustly conical, Theorem 1 applies (as if the columns of T were not outliers) and, for all $1 \leq k \leq r + t$,

$$X(k, k) \geq 1 - \frac{8\epsilon}{\kappa(1-\beta)(1-\epsilon)} \geq \frac{1}{2},$$

while $X(j, j) \leq \frac{8\epsilon}{\kappa(1-\beta)(1-\epsilon)} \leq \frac{1}{2}$ for all $j > r + t$, since $\epsilon \leq \frac{\nu(1-\beta)}{20(n-1)}$ where $\nu = \min(\kappa, \eta, \delta)$. Therefore, only the first $r + t$ indices can potentially be extracted by Algorithm 5. It remains to bound above (resp. below) the off-diagonal entries of the rows of X corresponding to T (resp. W).

By Lemma 2 (see also [9, Lemma 2]), we have for all $1 \leq j \leq n$

$$\|M(:, j) - MX(:, j)\|_1 \leq \frac{4\epsilon}{1-\epsilon} \quad \text{and} \quad \|X(:, j)\|_1 \leq 1 + \frac{4\epsilon}{1-\epsilon}.$$

Using the fact that $[W, T]$ is κ -robustly conical, for all $1 \leq k \leq t$, we have

$$\|T(:, k) - MX(:, k)\|_1 \geq (1 - X(k, k)) \min_{x \geq 0, y \geq 0} \|T(:, k) - T(:, \bar{K})x - Wy\|_1 \geq (1 - X(k, k))\kappa,$$

implying that for all $1 \leq k \leq t$

$$X(k, k) \geq 1 - \frac{4\epsilon}{\kappa(1-\epsilon)} \geq \frac{1}{2},$$

since $\frac{4}{1-\epsilon} \leq 5$ because $\epsilon \leq \frac{1}{20}$. Therefore,

$$\sum_{j \neq k} X(j, k) \leq \|X(:, k)\|_1 - X(k, k) \leq \frac{4\epsilon}{1-\epsilon} + \frac{4\epsilon}{\kappa(1-\epsilon)} \leq \frac{8\epsilon}{\kappa(1-\epsilon)},$$

as $\kappa, \epsilon \leq 1$. Let $t+1 \leq j \leq n$ and $1 \leq k \leq t$, we have

$$\|M(:,j) - MX(:,j)\|_1 \geq \min_x \min_{y \geq 0} \|T(:,k) + T(:,\bar{\mathcal{K}})y - Wx\|_1 \geq \eta X(k,j),$$

see Equation (5), which implies $X(k,j) \leq \frac{4\epsilon}{\eta(1-\epsilon)}$. Hence, for all $1 \leq k \leq t$, we have

$$\sum_{j \neq k} X(k,j) \leq (t-1) \frac{8\epsilon}{\kappa(1-\epsilon)} + (n-r-t) \frac{4\epsilon}{\eta(1-\epsilon)} \leq \frac{8(n-1)\epsilon}{\nu(1-\epsilon)} \leq \frac{1}{2}.$$

since $\nu = \min(\kappa, \eta, \delta)$. By assumption, for each $t+1 \leq k \leq t+r$, there exists some j satisfying $M(:,j) = WH(:,j) \neq W(:,k)$ and

$$\min_{x \geq 0} \|M(:,j) - W(:,\bar{\mathcal{K}})x\|_1 \geq \delta, \quad \text{where } \bar{\mathcal{K}} = \{1, 2, \dots, r\} \setminus \{k\},$$

see Equation (6). For $t+r < j \leq n$, we have $X(j,j) \leq \frac{8\epsilon}{\kappa(1-\beta)(1-\epsilon)}$. Let us denote $\mu = \frac{8(n-r-t)\epsilon}{\kappa(1-\beta)(1-\epsilon)}$ which is an upper bound for the total weight that can be assigned to the columns of M different from W and T . Then, using Equation (6), we have

$$\begin{aligned} \|M(:,j) - MX(:,j)\|_1 &\geq (1-\mu) \min_{y \geq 0} \left\| M(:,j) - \frac{1}{1-\mu} WX(t+1:r+t, j) - Ty \right\|_1 \\ &\geq (1-\mu) \left(1 - \frac{X(k,j)}{1-\mu} \right) \delta. \end{aligned}$$

This implies

$$\frac{X(k,j)}{1-\mu} \geq 1 - \frac{4\epsilon}{\delta(1-\mu)(1-\epsilon)}$$

and

$$\begin{aligned} X(k,j) &\geq 1 - \frac{8(n-r-t)\epsilon}{\kappa(1-\beta)(1-\epsilon)} - \frac{4\epsilon}{\delta(1-\epsilon)} \\ &\geq 1 - \frac{8(n-1)\epsilon}{\nu(1-\beta)(1-\epsilon)} \geq \frac{1}{2}, \end{aligned}$$

since $\beta \leq 1$ and $\epsilon \leq \frac{\nu(1-\beta)}{20(n-1)}$, and the proof is complete. \square

References

- [1] Araújo, U., Saldanha, B., Galvão, R., Yoneyama, T., Chame, H., Visani, V.: The successive projections algorithm for variable selection in spectroscopic multicomponent analysis. *Chemometrics and Intelligent Laboratory Systems* **57**(2), 65–73 (2001)
- [2] Arora, S., Ge, R., Kannan, R., Moitra, A.: Computing a nonnegative matrix factorization – provably. In: *Proceedings of the 44th symposium on Theory of Computing, STOC '12*, pp. 145–162 (2012)
- [3] Arora, S., Ge, R., Moitra, A.: Learning topic models - going beyond svd. In: *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS '12* (2012). To appear
- [4] Bittorf, V., Recht, B., Ré, E., Tropp, J.: Factoring nonnegative matrices with linear programs. In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 1223–1231 (2012)

- [5] Chan, T.H., Ma, W.K., Chi, C.Y., Wang, Y.: A convex analysis framework for blind separation of non-negative sources. *IEEE Trans. on Signal Processing* **56**(10), 5120–5134 (2008)
- [6] Chen, L., Choyke, P., Chan, T.H., Chi, C.Y., Wang, G., Wang, Y.: Tissue-specific compartmental analysis for dynamic contrast-enhanced mr imaging of complex tumors. *IEEE Trans. on Medical Imaging* **30**(12), 2044–2058 (2011)
- [7] Elhamifar, E., Sapiro, G., Vidal, R.: See all by looking at a few: Sparse modeling for finding representative objects. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2012)
- [8] Esser, E., Moller, M., Osher, S., Sapiro, G., Xin, J.: A convex model for nonnegative matrix factorization and dimensionality reduction on physical space. *IEEE Transactions on Image Processing* **21**(7), 3239–3252 (2012)
- [9] Gillis, N.: Robustness analysis of hotttopixx, a linear programming model for factoring nonnegative matrices (2012). [arXiv:1211.6687](https://arxiv.org/abs/1211.6687)
- [10] Gillis, N.: Sparse and unique nonnegative matrix factorization through data preprocessing. *Journal of Machine Learning Research* **13**(Nov), 3349–3386 (2012)
- [11] Gillis, N., Vavasis, S.: Fast and robust recursive algorithms for separable nonnegative matrix factorization (2012). [arXiv:1208.1237](https://arxiv.org/abs/1208.1237)
- [12] Glineur, F., Terlaky, T.: Conic formulation for l_p -norm optimization. *Journal of Optimization Theory and Applications* **122**(2), 285–307 (2004)
- [13] Kumar, A., Sindhwani, V., Kambadur, P.: Fast conical hull algorithms for near-separable non-negative matrix factorization. In: *International Conference on Machine Learning (ICML)* (2013)
- [14] Vavasis, S.: On the complexity of nonnegative matrix factorization. *SIAM Journal on Optimization* **20**(3), 1364–1377 (2009)